**Artificial Intelligence Practical**

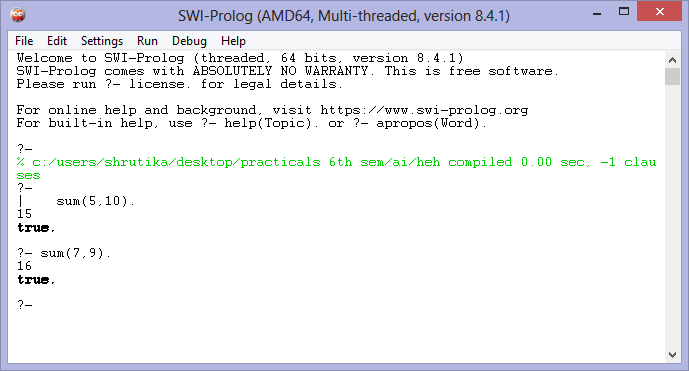
**Name: Shrutika Shaw | Roll no.: 2019345**

1. **Write a prolog program to calculate sum of two numbers.**

**Program:**

sum(X,Y):- S is X+Y, write(S).

Output:



1. **Write a prolog program to implement max(X,Y,M) where M is the maximum of the two numbers X and Y.**

**Program:**

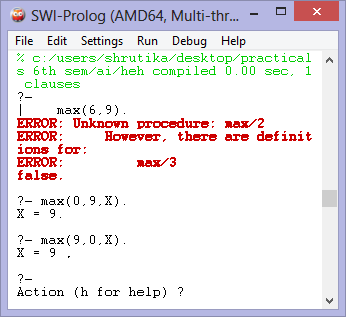
max(X,Y,M):-X>Y,

M is X.

max(X,Y,M):-Y>=X,

M is Y.

**Output:**



1. **Write a prolog program to implement factorial(N,F) where F is the factorial of number N.**

**Program:**

factorial(0,1).

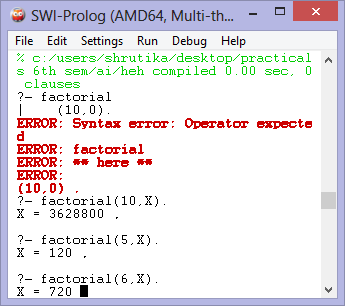
factorial(N,F):-

N >=1,

N1 is N-1,

factorial(N1,F1),

F is N\*F1.



1. **Write a prolog program to implement generate\_fib(N,T) where T represents the Nth term of the fibonacci series.**

**Program:**

fibonacci(1,0).

fibonacci(2,1).

fibonacci(N,X):-

N > 2,

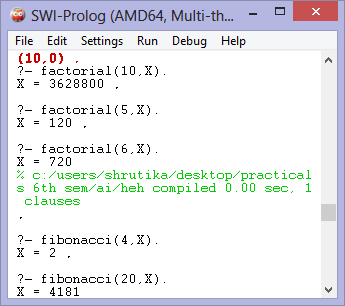
N1 is N-1,

N2 is N-2,

fibonacci(N1,X1),

fibonacci(N2,X2),

X is X1+X2.



1. **Write a prolog program to implement GCD of two numbers.**

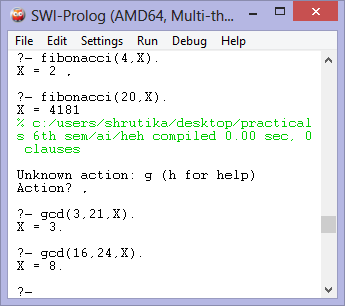
**Program:**

gcd(0,A,A):-!.

gcd(A,0,A):-!.

gcd(A,B,R):-B1 is mod(A,B),

gcd(B,B1,R).

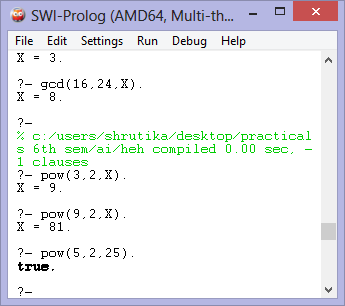


1. **Write a prolog program to implement power(Num,Pow,Ans): where Num is raised to the Pow to get.**

**Program:**

power(\_,0):-!.

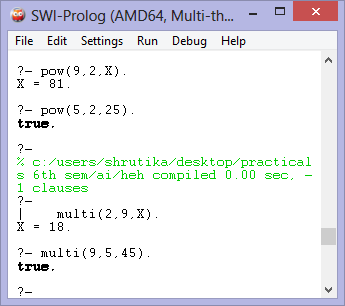
power(Num,Pow,Ans):-Ans is Num^Pow.



1. **Write a prolog program to implement multi(N1,N2,R): where N1 and N2 denote the numbers to be multiplied and R represents the result.**

**Program:**

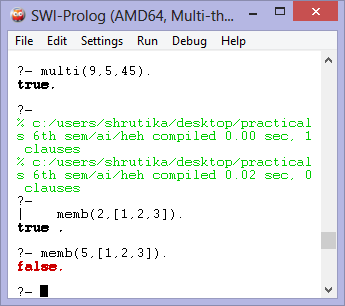
multi(N1,N2,R):-R is N1\*N2.



1. **Write a prolog program to implement memb(X,L): to check whether X is a member of L or not.**

**Program:**

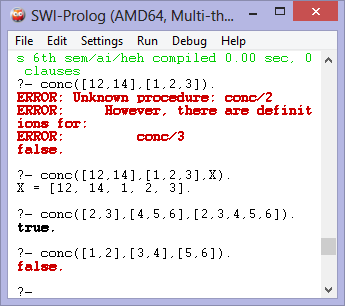
memb(X,[X|\_]). memb(X,[\_|Z]):- memb(X,Z).



1. **Write a Prolog program to implement conc (L1, L2, L3) where L2 is the list to be appended with L1 to get the resulted list L3.**

**Program**:

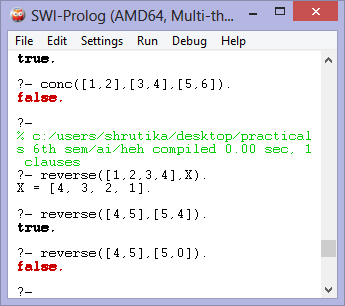
conc([],L,L). conc([X|L1],L2,[X|L3]):-conc(L1,L2,L3).



1. **Write a prolog program to implement reverse (L,R) where List L is original and list R is reversed list.**

**Program:**

ar([H|T],A,R):-ar(T,[H|A],R). ar([],A,A). reverse(L,R):-ar(L,[],R).



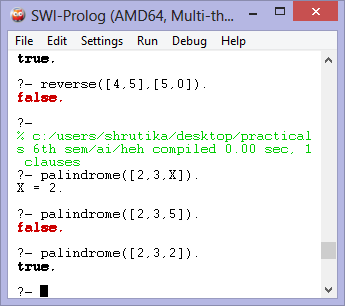
1. **Write a prolog program Implement palindrome(L) to check whether list L is a palindrome or not.**

**Program:**

ar([H|T],A,R):-ar(T,[H|A],R).

ar([],A,A). rev(L,R):-ar(L,[],R).

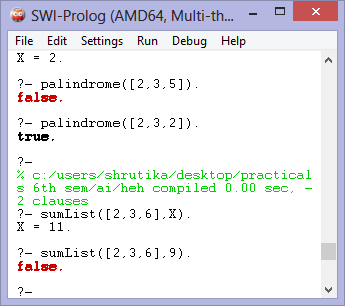
palindrome(L):-rev(L,L).



1. **Write a Prolog program to implement sumlist(L, S) so that S is the sum of a given list L.**

**Program:**

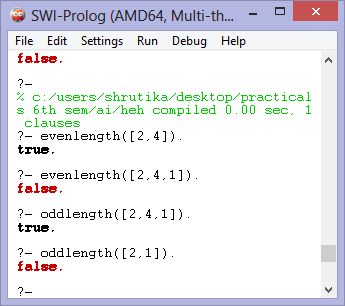
sumList([],0). sumList([H|T],S):-sumList(T,S1), S is H + S1.

****

**13. Write a Prolog program to implement two predicates evenlength(List) and oddlength(List) so that they are true if their argument is a list of even or odd length respectively.**

**Program:**

evenlength([]). evenlength([\_|R]):- oddlength(R). oddlength([\_|R]):- evenlength(R).

****

**14. Write a prolog program to implement nth\_element(N,L,X) where N is desired position in a list and X represents the Nth element of L.**

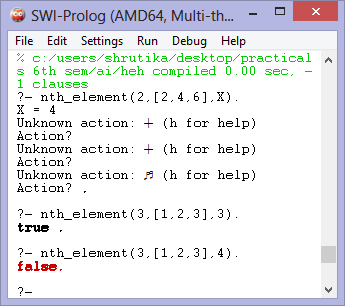
**Program:**

nth\_element(1,[H|\_],H).

nth\_element(N,[\_|T],X):-

N1 is N-1,

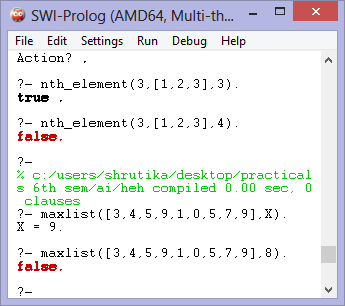
nth\_element(N1,T,X).

****

**15. Write a Prolog program to implement maxlist(L, M) so that M is the maximum number in the list.**

**Program:**

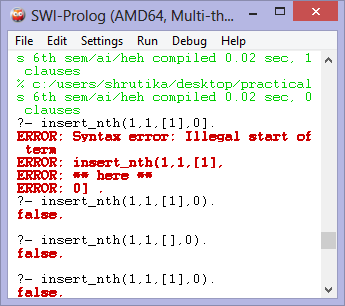
maxlist([H],H). maxlist([H|T],M):- maxlist(T,M1), H M is M1;M is H.

****

**16. Write a prolog program to implement insert\_nth(I, N, L, R) that inserts an item I into Nth position of list L to generate a list R.**

**Program:**

insert\_nth(I,1,L,[I|L]). insert\_nth(I,N,[X|Y],[X|Z]):-A is N1,insert\_nth(I,A,Y,Z).

****

**17. Write a prolog program to Implement delete\_nth(N,L,R) to remove Nth element from list L to generate list R.**

**Program:**

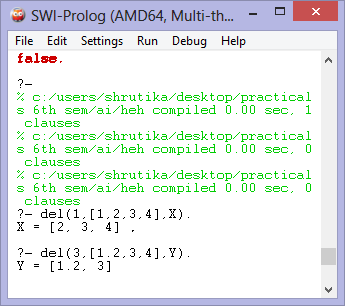
del(1,[\_|T],T).

del(\_,[],[]).

del(N,[H|T],[H|T1]):-

N1 is N-1,

del(N1,T,T1).



**18. Write a program in prolog to implement merge(L1,L2,L3) where L1 is first ordered list and L2 is a second order list and L3 represent the merged list.**

**Program:**

merge\_list([],[],[]). merge\_list([],L2,L2). merge\_list(L1,[],L1). merge\_list([H1|T1],[H2|T2],[H1|T3]):- H1= merge\_list(T1, [H2|T2], T3). merge\_list([H1|T1],[H2|T2],[H2|T3]):- merge\_list([H1|T1], T2, T3).

